



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

«Разработка приложения для хранения и анализа информации о киберспортивных матчах и командах»

Студент ИУ7-66Б
(Группа)

(Подпись, дата)

Леонов В. В.
(И. О. Фамилия)

Руководитель курсовой работы

(Подпись, дата)

Кивва К. А.
(И. О. Фамилия)

2022 г.

РЕФЕРАТ

Расчетно-пояснительная записка 45 с., 16 рис., 4 табл., 11 источн., 4 прил.

Ключевые слова: Базы Данных, PostgreSQL, Golang, реляционная модель данных, киберспорт, Dota 2.

Целью курсовой работы является проектирование и реализация базы данных, содержащей информацию о киберспортивных матчах и командах в дисциплине Dota 2.

Для достижения указанной цели были выполнены следующие задачи:

- рассмотреть существующие сервисы;
- формализовать задание и определить необходимый функционал;
- спроектировать базу данных, описать ее сущности и связи;
- выбрать подходящую систему управления базами данных;
- реализовать базу данных;
- провести сравнительный анализ времени обработки запроса в базе данных с индексацией и без;
- реализовать программное обеспечение, которое позволит получить доступ к данным.

Цель курсовой работы была достигнута в полном объеме: была изучена предметная область, были выполнены проектирование и реализация базы данных, содержащей информацию о киберспортивных матчах и командах в дисциплине Dota 2, были созданы соответствующие триггеры и проведено исследование влияния индексации записей на скорость выполнения запросов в базе данных.

СОДЕРЖАНИЕ

РЕФЕРАТ	3
ВВЕДЕНИЕ	6
1 Аналитический раздел	7
1.1 Постановка задачи	7
1.2 Актуальность проекта	7
1.3 Предметная область	7
1.3.1 Краткое описание игрового процесса	7
1.3.2 Dota Pro Circuit	8
1.4 Обзор существующих сервисов	8
1.5 Типы пользователей	11
1.6 Формализация данных	13
1.7 Выводы	15
2 Конструкторский раздел	16
2.1 Схемы базы данных	16
2.2 Схемы триггеров	23
2.3 Выводы	23
3 Технологический раздел	24
3.1 Выбор технологий разработки	24
3.2 Методология построения ПО	24
3.3 Описание интерфейса программного продукта	26
4 Исследовательский раздел	30
ЗАКЛЮЧЕНИЕ	32
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	33
ПРИЛОЖЕНИЕ А Создание базы данных	34
ПРИЛОЖЕНИЕ Б Ограничения в базе данных	37
ПРИЛОЖЕНИЕ В Ролевая модель	41

ВВЕДЕНИЕ

Сегодня, в век высоких технологий, наравне с общеизвестными спортивными дисциплинами звучит слово «киберспорт» и привлекает интерес большого количества людей. Как и в любом виде спорта в киберспорте проводится множество турниров различного уровня во всех регионах с обширным количеством игроков, спонсоров и матчей внутри конкретного соревнования. Одной из самых популярных дисциплин и с огромным количеством зрителей является Dota 2 [1].

Многие люди имеют любимые команды, фаворитов среди игроков и следят за их карьерой. Таким образом, существует потребность в сервисе, предоставляющем данные о турнирах, командах, внутриматчевую или глобальную статистику выступлений и другую сопутствующую информацию.

Целью курсовой работы является проектирование и реализация базы данных, содержащей информацию о киберспортивных матчах и командах в дисциплине Dota 2.

Для достижения указанной выше цели следует выполнить следующие задачи:

- рассмотреть существующие сервисы;
- формализовать задание и определить необходимый функционал;
- спроектировать базу данных, описать ее сущности и связи;
- выбрать подходящую систему управления базами данных;
- реализовать базу данных;
- провести сравнительный анализ времени обработки запроса в базе данных с индексацией и без;
- реализовать программное обеспечение, которое позволит получить доступ к данным.

1 Аналитический раздел

В аналитическом разделе выполнено изучение предметной области, представлено формальное описание данных, выполнен анализ существующих моделей и методов организации хранения данных в базе знаний.

1.1 Постановка задачи

Необходимо спроектировать и реализовать базу данных, содержащую информацию о киберспортивных матчах и командах в дисциплине Dota 2. Следует разработать интерфейс, который позволит работать с этой базой данных: добавлять, удалять и редактировать информацию, составлять отчеты по периодам, задаваемым пользователем, с их графическим представлением. Требуется предусмотреть наличие ролей модератора и администратора, осуществляющих контроль за добавлением новых турниров, регистрацией новых команд и обновлением текущих составов.

1.2 Актуальность проекта

С каждым годом призовые фонды турниров становятся все больше. Более того, киберспортивные первенства проходят даже в пределах конкретных организаций, школ, колледжей, университетов и других объединений, которые в свою очередь нельзя подробно изучить в одном месте.

Российская команда Team Spirit стала победителем The International 10, что существенным образом сказалось на популярности и интересе к киберспортивной составляющей Dota 2 [2].

Таким образом, разрабатываемое приложение предложит удобный инструмент для просмотра статистики по киберспортивным турнирам, с возможностями добавления своих и составления отчетов за выбранный период.

1.3 Предметная область

1.3.1 Краткое описание игрового процесса

Dota 2 представляет собой соревновательную командную игру с RPG (англ. Role-Playing Game — ролевая игра) элементами типа МОБА («многопользовательская онлайн-боевая арена»). Две соревнующиеся команды Света (англ. The Radiant) и Тьмы (англ. The Dire) состоят из пяти игроков

каждая. Главной целью игры в Dota 2 является уничтожение вражеской крепости (англ. Ancient). Каждая из крепостей защищается несколькими башнями, расположенными вдоль трех линий обороны. Вместо построения собственной армии как в классических RTS играх (англ. Real Time Strategy — Стратегии реального времени), каждый игрок управляет одним уникальным героем с отличительными умениями, узнаваемой внешностью, силами, слабостями и уникальным стилем игры [3].

Каждый из игроков выполняет определенную роль на протяжении матча, как например «керри» (англ. Carry) или «саппорт» (англ. Support), и герои могут по своим характеристикам лучше подходить для той или иной роли. Во время игры герой имеет возможность зарабатывать опыт и золото, которое можно тратить на покупку предметов, которые позволяют усилить одну или несколько характеристик персонажа: урон по другим героям, восстановление здоровья или потенциал уничтожения вражеских строений.

Если очки здоровья героя снижаются до нуля — например, его одолевает в бою вражеский герой — герой в течение некоторого короткого времени считается «погибшим»; по окончании этого времени герой вновь появляется рядом с крепостью, и управляющий им игрок может возобновить игру. Убийство врага позволяет получить большое количество дополнительного золота и опыта.

1.3.2 Dota Pro Circuit

Dota Pro Circuit — профессиональные турниры организуются компанией Valve и её партнёрами, для определения команд, которые получают прямые приглашения на The International (главный турнир года).

Заработанные за сезон баллы будут определять статус команды перед The International и её шансы на получение прямого приглашения [4].

1.4 Обзор существующих сервисов

Следует отметить, что существуют сервисы, которые агрегируют, систематизируют и визуализируют информацию о киберспортивных турнирах.

DOTABUFF является крупнейшим сервисом для сбора и анализа матчевой статистики, в том числе о профессиональных матчах. Однако, данный сервис доступен только на английском языке и помимо киберспортивной информации имеет множество дополнительной, которая несколько осложняет

взаимодействие (см. рисунок 1.1).

The screenshot shows the Dota Buff website interface. At the top, there's a navigation bar with links like 'Download', 'Esports', 'Heroes', 'Items', 'Players', 'Matches', 'Blog', 'Forums', and 'Plus'. Below this, a banner for the 'ESL One Stockholm Major 2022 powered by Intel' is displayed, indicating a 'Premium League' with a '\$500,000 TOTAL PRIZE POOL'. The main content area is divided into three sections:

- TEAMS:** A table listing teams and their statistics.

Team	Series	Matches	Heroes	KDA	GPM	XPM
Tundra Esports	6-1-0	13-2	51	5.85	2,398	2,921
Thunder Predator	5-3-1	10-6	50	4.62	2,352	3,013
Gaimin Gladiators	3-2-1	8-4	51	3.85	2,269	2,811
TSM FTX	4-2-2	9-6	65	3.86	2,319	2,850
BetBoom Team	2-3-2	8-7	55	3.97	2,223	2,768
OG	3-1-3	6-7	54	2.86	2,250	2,816
T1	2-2-3	6-8	51	3.41	2,140	2,628
Team Spirit	3-1-5	7-9	56	2.58	2,269	2,766
- LIVE MATCHES ON TRACKDOTA:** A section indicating 'No matches are currently being played.'
- RECENT SERIES:** A table showing recent match results.

Series	Details	Winner	Teams	Games
B03 Russia	Completed 10 hours ago	2-0 GG	Gaimin Gladiators vs T1	1 2 0
B03 Russia	Completed 12 hours ago	2-0 TSM FTX	TSM FTX vs OG	1 2 0
B03 Russia	Completed 15 hours ago	2-1 ThunderP	Thunder Predator vs BetBoom Team	1 2 3
B03 Russia	Completed 19 hours ago	2-1 Tundra	Tundra Esports vs Team Spirit	1 2 3
- SUCCESSFUL PLAYERS:** A table listing top players and their statistics.

Player	Matches	KDA
33 Tundra	13-2	8.50
skiter Tundra	13-2	7.56
Pakazs ThunderP	10-6	8.87
Nine Tundra	13-2	5.92
Saksa Tundra	13-2	4.78
Sneyking Tundra	13-2	4.42
BOOM GG	8-4	7.00

Рисунок 1.1 – Портал dotabuff.com

OpenDota — это проект энтузиастов с открытым исходным кодом, собирающий данные Dota 2. Сервис предоставляет веб-интерфейс для обычных пользователей, так же как API для разработчиков с возможностью интеграции в сторонние приложения. Большим плюсом является возможность детального изучения матча с автоматизированной аналитикой и советами (см. рисунок 1.2) [5].

Radiant Victory 34 ALL DRAFT 24:24 7 ENDED 15 HOURS AGO

MATCH ID: 6573204421 REGION: Stockholm

⚠ The replay for this match has not yet been parsed. Not all data may be available.

PARSE REPLAY GET FREE DIGITAL COACH

OVERVIEW BENCHMARKS PERFORMANCES LANING COMBAT FARM ITEMS GRAPHS CASTS OBJECTIVES VISION ACTIONS TEAMFIGHTS ANALYSIS COSMETICS LOG FANTASY CHAT STORY

Radiant - Overview WINNER

PLAYER	LVL	K	D	A	LI/DN	NET	GPM/XPM	HD	TD	HH	ITEMS	BUFFS
Anonymous 1. Unknown	16	8	1	13	79/10	10.1k	447/538	17k	4.4k	-		
v zagon 2. Divine [5]	19	11	2	10	163/3	14.4k	508/722	13.8k	8.4k	220		
BLAME 3. Immortal	17	4	2	6	154/8	12.2k	510/559	7.5k	6.1k	-		
Монголой пан... 4. Divine [5]	13	5	3	13	60/5	6.6k	381/504	11.6k	1.7k	1.1k		
Reinard12 5. Divine [5]	15	5	1	16	35/2	7.6k	374/502	10.6k	2.1k	2.2k		
TOTAL	82	33	7	58	511/28	52.8k	2.3k/2.8k	60.5k	22.7k	3.5k		

PICK 1 PICK 3 PICK 5 PICK 7 PICK 9
BAN 1 BAN 2 BAN 3 BAN 4

Рисунок 1.2 – Портал.opendota.com

На рисунке 1.3 представлен портал cybersport.ru. Данный сервис специализируется на широком круге киберспортивных дисциплин. Портал позволяет смотреть матчи в прямом эфире, следить за турнирной сеткой, читать новости о профессиональных игроках и командах, но не дает пользователю подробной статистики и послематчевой аналитики. Доступность на русском языке выгодно выделяет сервис на фоне конкурентов.

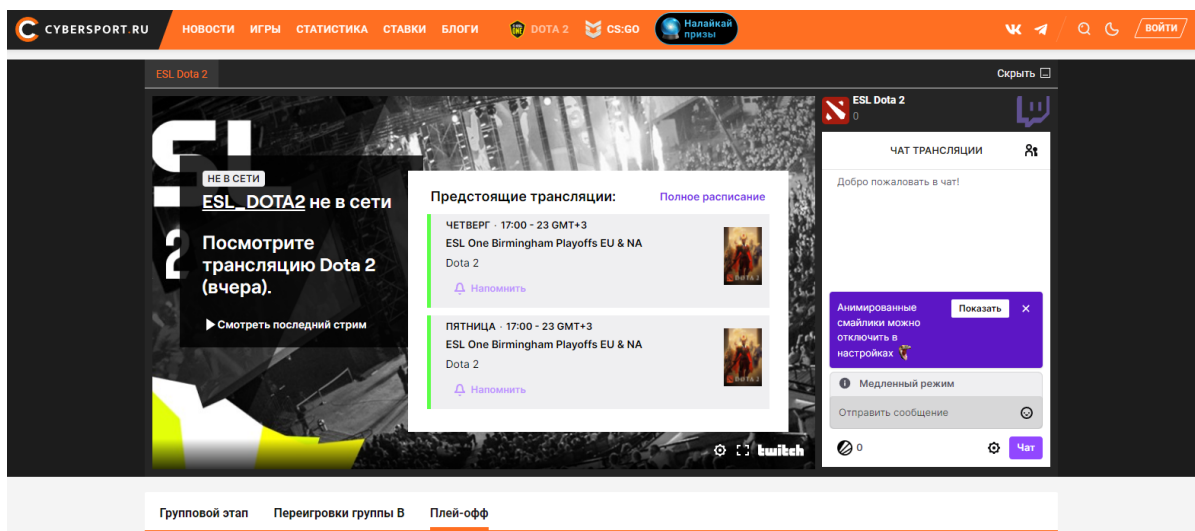


Рисунок 1.3 – Портал cybersport.ru

Таким образом, на основе рассмотренных сервисов можно составить сравнительную таблицу 1.1.

Ни один из представленных сервисов не обладает desktop-приложением, возможностью просмотра турниров и доступностью на русском языке. Разрабатываемое ПО должно соответствовать этим требованиям.

Таблица 1.1 – Сравнительная таблица сервисов

	DOTABUFF	OpenDota	cybersport.ru
Открытый исход- ный код	-	+	-
Доступность	По подписке	Бесплатно	С рекламой
Послематчевая ста- тистика	+	+	-
Турниры	-	-	+
Язык	Английский	Английский	Русский
Наличие desktop- приложения	+	-	-

1.5 Типы пользователей

Разрабатываемая система ориентирована на многопользовательское использование, следовательно, важным является разделение пользователей по ролям и соответствующему им функционалу (таблица 1.2).

Таблица 1.2 – Типы пользователей и доступный им функционал

Тип пользователя	Доступный функционал
Неавторизованный	Регистрация, авторизация
Клиент	Просмотр информации о киберспортивных матчах, прошедших турнирах, составах команд, сведений о игроках и сводной статистики за заданный период
Модератор	Просмотр информации о киберспортивных матчах, прошедших турнирах, составах команд, сведений о игроках и сводной статистики за заданный период Операции доступа, изменения и удаления информации в базе знаний
Администратор	Просмотр информации о киберспортивных матчах, прошедших турнирах, составах команд, сведений о игроках и сводной статистики за заданный период Операции доступа, изменения и удаления информации в базе знаний Добавление и удаление пользователей, изменение типа существующего пользователя

Возможные варианты взаимодействия с системой описаны при помощи

UseCase-диаграммы, которая приведена на рисунке 1.4.

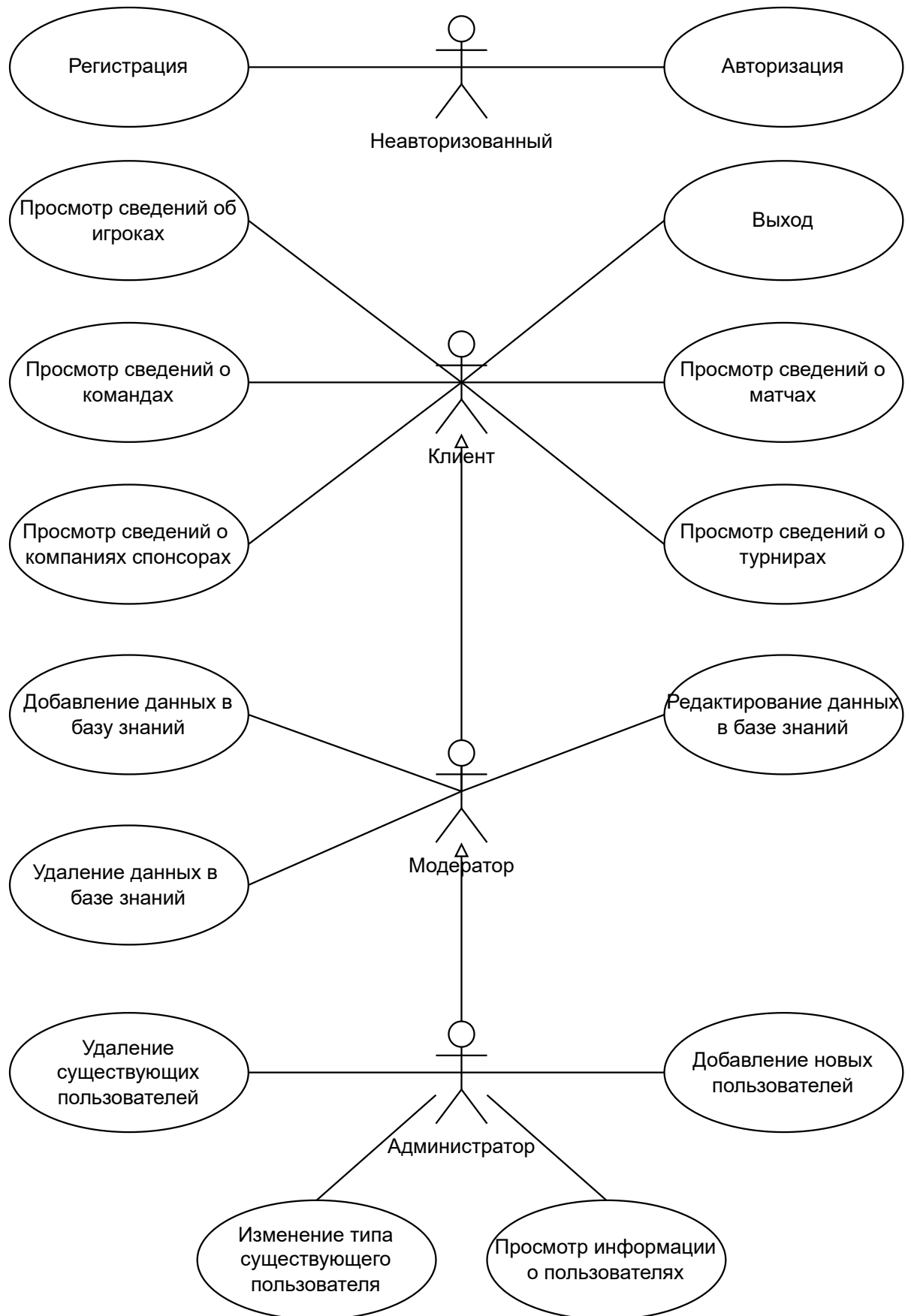


Рисунок 1.4 – UseCase-диаграмма

1.6 Формализация данных

База данных должна хранить информацию о:

- турнирах;
- компаниях-спонсорах;
- командах;
- игроках;
- матчах;
- пользователях.

В таблице 1.3 описано формальное представление данных и сведения, которые они содержат.

Таблица 1.3 – Данные и сведения о них

Данные	Сведения
Турнир	Название, уровень, призовой фонд, дата, время и место проведения, количество DPC очков
Компания-спонсор	Название, страна, вебсайт, ежегодная выручка, область деятельности
Команда	Название, дата создания, контактный e-mail, общий заработок, регион, уровень
Игрок	Псевдоним, имя, дата рождения, страна, личный рейтинг, игровая позиция, сигнатурный герой
Матч	Продолжительность, победитель, герои, соотношение убийств, смертей и помощи, общая ценность, опыт и золото в минуту, урон
Пользователь	Логин, пароль, права доступа, e-mail, имя

На основе выделенных данных и сведений, которые они содержат, следует составить диаграмму в нотации Чена, описывающую сущности исследуемой системы, а также их взаимодействие (рисунок 1.5).

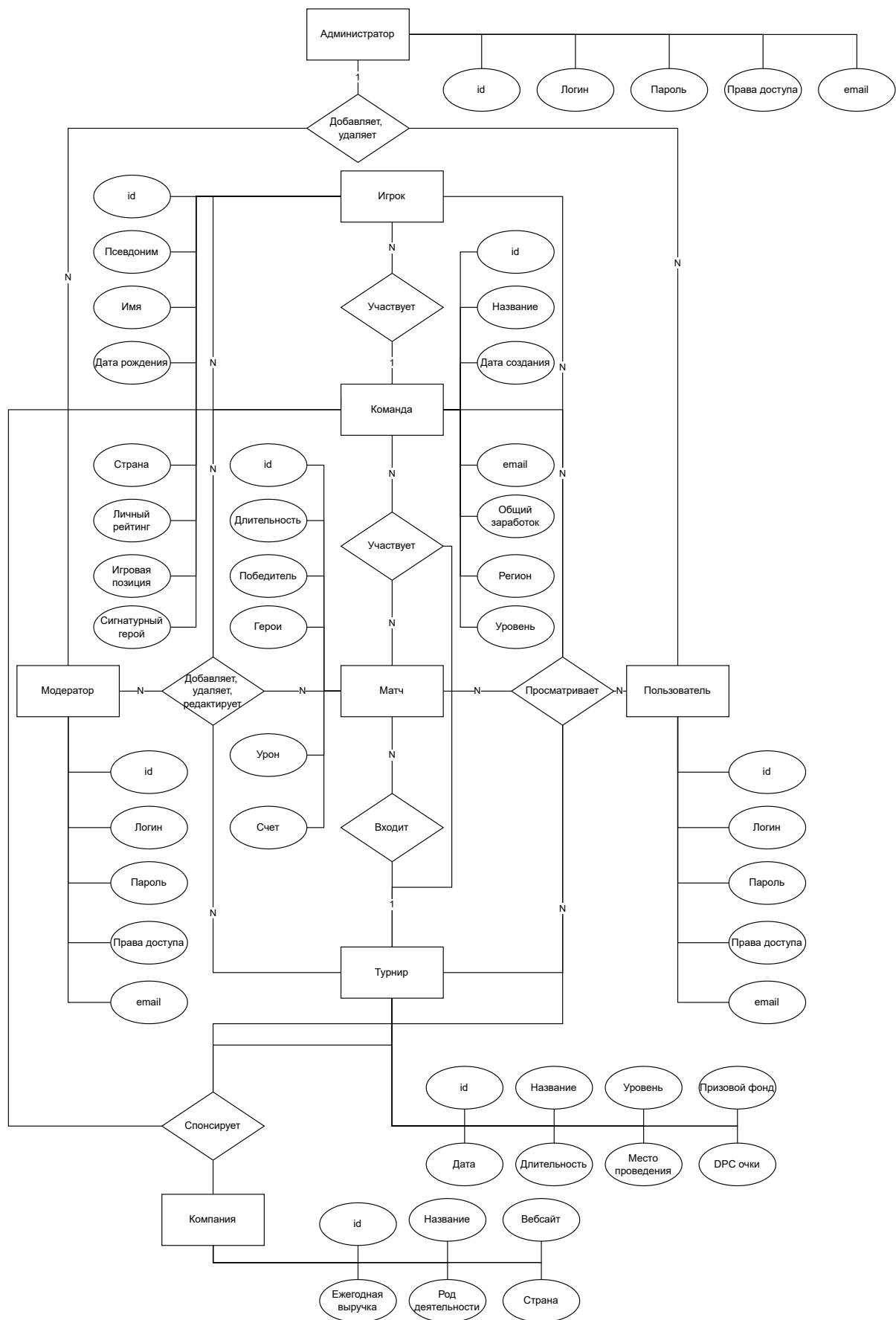


Рисунок 1.5 – Диаграмма сущность–связь

1.7 Выводы

В данном разделе было выполнено изучение предметной области, представлено формальное описание данных, возможные варианты взаимодействия с системой описаны при помощи UseCase-диаграммы, а также составлена диаграмма сущность-связь предметной области.

2 Конструкторский раздел

2.1 Схемы базы данных

База данных должна включать следующие таблицы:

- таблица о турнирах — **Tournaments**;
- таблица о компаниях-спонсорах — **Companies**;
- таблица о командах — **Teams**;
- таблица о игроках — **Players**;
- таблица о матчах — **Matches**;
- таблица о пользователях — **Users**.

Далее представлено подробное описание полей, каждой из таблиц. Диаграмма базы данных изображена на рисунке 2.1.

Таблица **Tournaments** описывает турниры и содержит следующие поля:

- **id** — идентификатор турнира (первичный ключ), представляется целым числом;
- **name** — название турнира, представляется строкой;
- **tier** — уровень престижности турнира, представляется целым числом;
- **prize_pool** — призовой фонд турнира, представляется целым числом;
- **data_start** — дата начала проведения турнира, представляется типом `date`;
- **duration** — продолжительность турнира в днях, представляется целым числом;
- **dpc_points** — количество DPC-очков на турнире, представляется целым числом;
- **location** — место проведения турнира, представляется строкой.

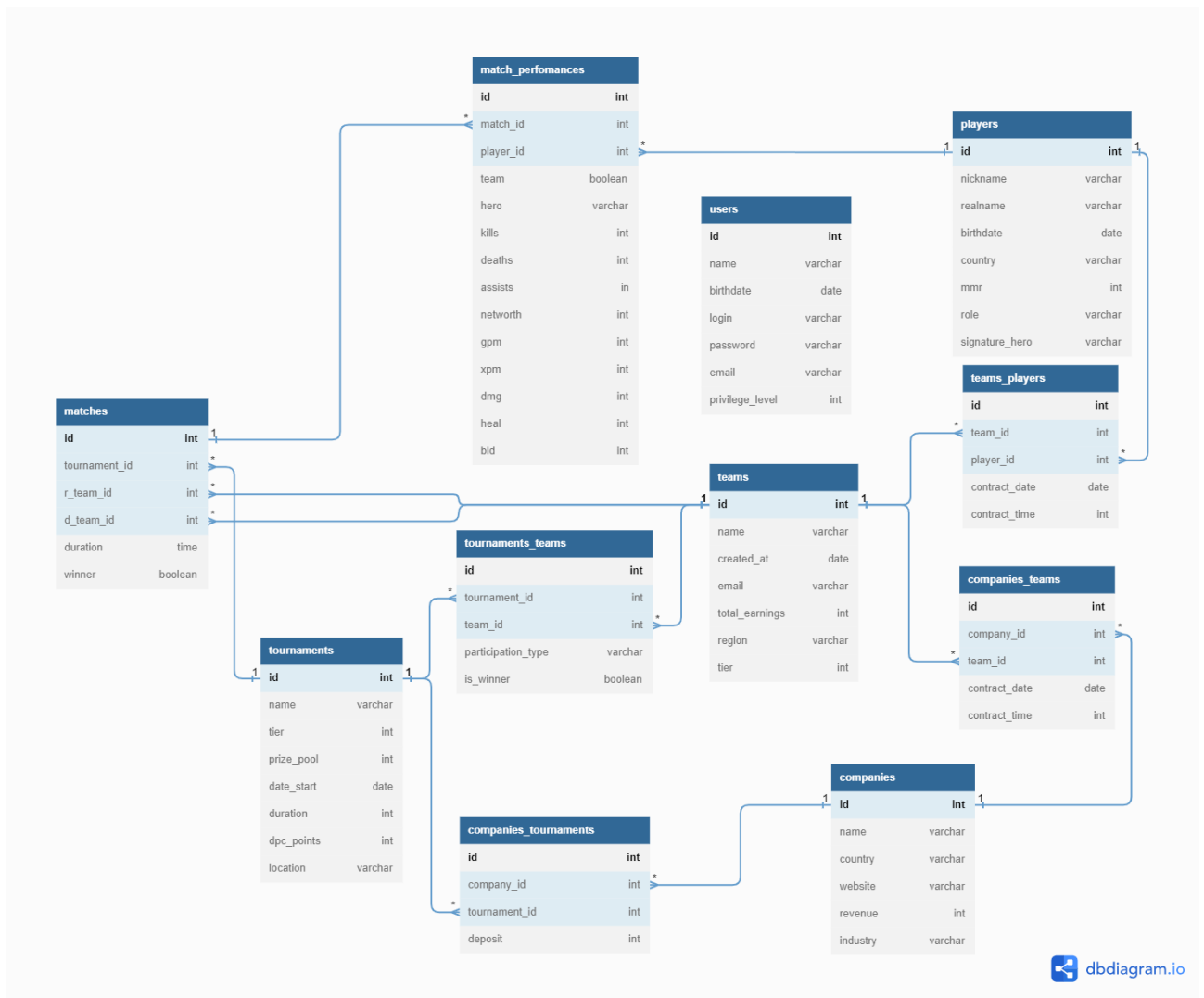


Рисунок 2.1 – Диаграмма базы данных

Таблица **Companies** описывает компании-спонсоры и содержит следующие поля:

- **id** — идентификатор компании (первичный ключ), представляется целым числом;
- **name** — название компании, представляется строкой;
- **country** — страна компании, представляется строкой;
- **website** — вебсайт компании, представляется строкой;
- **revenue** — годовой оборот компании, представляется целым типом;
- **industry** — сфера деятельности компании.

Таблица **Teams** описывает команды и содержит следующие поля:

- **id** — идентификатор команды (первичный ключ), представляется целым числом;
- **name** — название команды, представляется строкой;
- **created_at** — дата создания команды, представляется типом дата;
- **email** — email команды, представляется строкой;
- **total_earnings** — общий заработок команды, представляется целым типом;
- **region** — регион, который представляет команда, представляется строкой;
- **tier** — уровень команды, представляется целым типом.

Таблица **Players** описывает игроков и содержит следующие поля:

- **id** — идентификатор игрока (первичный ключ), представляется целым числом;
- **nickname** — псевдоним игрока, представляется строкой;
- **realname** — имя игрока, представляется строкой;
- **birthdate** — дата рождения игрока, представляется типом дата;
- **country** — страна игрока, представляется строкой;
- **mmr** — личный рейтинг игрока, представляется целым числом;
- **role** — избранная роль игрока, представляется строкой;
- **signature_hero** — избранный герой игрока, представляется строкой.

Таблица **Matches** описывает матчи и содержит следующие поля:

- **id** — идентификатор матча (первичный ключ), представляется целым числом;

- **tournament_id** — идентификатор турнира (является ссылкой на таблицу **Tournaments**), представляется целым числом;
- **r_team_id** — идентификатор команды Света (является ссылкой на таблицу **Teams**), представляется целым числом;
- **d_team_id** — идентификатор команды Тьмы (является ссылкой на таблицу **Teams**), представляется целым числом;
- **duration** — продолжительность матча, представляется целым числом;
- **winner** — победитель матча, представляется логическим типом.

Таблица **Users** описывает пользователей и содержит следующие поля:

- **id** — идентификатор пользователя (первичный ключ), представляется целым числом;
- **name** — имя пользователя, представляется строкой;
- **birthdate** — дата рождения пользователя, представляется типом **date**;
- **login** — логин пользователя, представляется строкой;
- **password** — пароль пользователя, представляется строкой;
- **email** — email пользователя, представляется строкой;
- **privilege_level** — уровень привилегий пользователя, представляется целым числом.

Таблицы **Tournaments-Teams** образуют связь многие-ко-многим. Необходимо создать развязочную таблицу **Tournaments_Teams**, которая будет содержать следующие поля:

- **id** — идентификатор записи (первичный ключ), представляется целым числом;
- **tournament_id** — идентификатор турнира (является ссылкой на таблицу **Tournaments**), представляется целым числом;

- `team_id` — идентификатор команды (является ссылкой на таблицу `Teams`), представляется целым числом;
- `participation_type` — способ отбора на турнир, представляется строкой;
- `is_winner` — является ли команда победителем, представляется логическим типом.

Таблицы `Companies-Tournaments` образуют связь многие-ко-многим. Необходимо создать развязочную таблицу `Companies_Tournaments`, которая будет содержать следующие поля:

- `id` — идентификатор записи (первичный ключ), представляется целым числом;
- `company_id` — идентификатор компании (является ссылкой на таблицу `Companies`), представляется целым числом;
- `tournament_id` — идентификатор турнира (является ссылкой на таблицу `Tournaments`), представляется целым числом;
- `deposit` — величина инвестиции компании, представляется целым числом.

Таблицы `Companies-Teams` образуют связь многие-ко-многим. Необходимо создать развязочную таблицу `Companies_Teams`, которая будет содержать следующие поля:

- `id` — идентификатор записи (первичный ключ), представляется целым числом;
- `company_id` — идентификатор компании (является ссылкой на таблицу `Companies`), представляется целым числом;
- `team_id` — идентификатор команды (является ссылкой на таблицу `Teams`), представляется целым числом;
- `contract_date` — дата подписания контракта, представляется типом `дата`;

- `contract_time` — срок контракта в месяцах, представляется целым числом.

Таблицы **Teams-Players** образуют связь многие-ко-многим. Необходимо создать развязочную таблицу **Teams_Players**, которая будет содержать следующие поля:

- `id` — идентификатор записи (первичный ключ), представляется целым числом;
- `team_id` — идентификатор команды (является ссылкой на таблицу **Teams**), представляется целым числом;
- `player_id` — идентификатор игрока (является ссылкой на таблицу **Players**), представляется целым числом;
- `contract_date` — дата подписания контракта, представляется типом `дата`;
- `contract_time` — срок контракта в месяцах, представляется целым числом.

Таблицы **Matches-Players** образуют связь многие-ко-многим. Необходимо создать развязочную таблицу **Match_Performances**, которая будет содержать следующие поля:

- **id** — идентификатор записи (первичный ключ), представляется целым числом;
- **match_id** — идентификатор матча (является ссылкой на таблицу **Matches**), представляется целым числом;
- **player_id** — идентификатор игрока (является ссылкой на таблицу **Players**), представляется целым числом;
- **team** — сторона (Свети или Тьма), представляется логическим типом;
- **hero** — герой, представляется строкой;
- **kills** — количество убийств, представляется целым числом;
- **deaths** — количество смертей, представляется целым числом;
- **assists** — количество помощей, представляется целым числом;
- **networth** — общая ценность, представляется целым числом;
- **gpm** — золото в минуту, представляется целым числом;
- **xpm** — опыта в минуту, представляется целым числом;
- **dmg** — общий урон по героям, представляется целым числом;
- **heal** — общее лечение, представляется целым числом;
- **bld** — общий по строениям, представляется целым числом.

2.2 Схемы триггеров

При заключении нового контракта на спонсорство турнира одной из компаний следует увеличивать призовой фонд соответствующего турнира. Для этого используется триггер, схема которого приведена на рисунке 2.2.



Рисунок 2.2 – Схема триггера обновления призового фонда турнира

При добавлении записи о победе команды в рамках определенного турнира следует увеличивать величину общего заработка команды. Для этого используется триггер, схема которого приведена на рисунке 2.3.

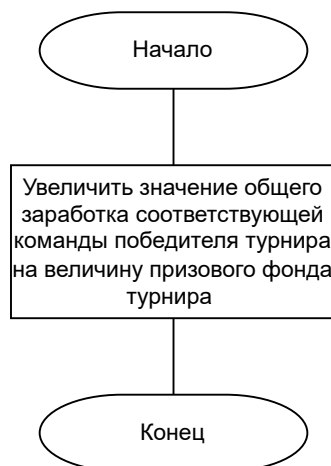


Рисунок 2.3 – Схема триггера обновления общего заработка команды

2.3 Выводы

В данном разделе было выполнено проектирование базы данных, описаны таблицы, которые необходимо реализовать в рамках рассматриваемой задачи, а также разработаны схемы триггеров для базы данных.

3 Технологический раздел

Далее в технологическом разделе будут описаны выбранные технологии разработки, методология построения ПО, а также рассмотрено взаимодействие пользователя с приложением.

3.1 Выбор технологий разработки

В качестве СУБД был сделан выбор в пользу PostgreSQL — это мощная система объектно-реляционных баз данных с открытым исходным кодом, которая использует и расширяет язык SQL в сочетании со многими функциями, позволяющими безопасно хранить и масштабировать самые сложные рабочие нагрузки данных [6].

В качестве ЯП был сделан выбор в пользу Go — это проект с открытым исходным кодом, призванный повысить продуктивность программистов. Язык Go выразительный, лаконичный, чистый и эффективный. Его механизмы параллелизма упрощают написание программ, максимально использующих многоядерные и сетевые машины, а новая система типов обеспечивает гибкое и модульное построение программ. Go быстро компилируется в машинный код, но обладает удобством сборки мусора и возможностями отражения во время выполнения. Это быстрый, статически типизированный компилируемый язык, который выглядит как динамически типизированный интерпретируемый язык [7].

Для разработки программного интерфейса используется Fyne — это простая в освоении бесплатная платформа с открытым исходным кодом для создания графических приложений для настольных компьютеров, мобильных устройств и других устройств. Сочетая мощь и простоту языка программирования Go с тщательно продуманной библиотекой виджетов разработчику легко создавать приложения и развертывать их [8].

3.2 Методология построения ПО

Разрабатываемое приложение спроектировано на основе чистой архитектуры. Этот способ организации кода подразумевает четкое строгое разделение ответственности. Приложение разбивается на независимые функциональные компоненты, которые взаимодействуют друг с другом определенным образом,

при этом между ними передаются только те ресурсы, которые необходимы для выполнения конкретной задачи. Это помогает минимизировать сложность каждого компонента, снизить вероятность ошибок и ускоряет их выявление [9].

На рисунке 3.1 представлено высокоуровневое разбиение приложения на компоненты. Данный подход позволяет легко масштабировать приложение, а также выполнять подмену независимых компонентов, не привязываясь к их конкретной реализации.

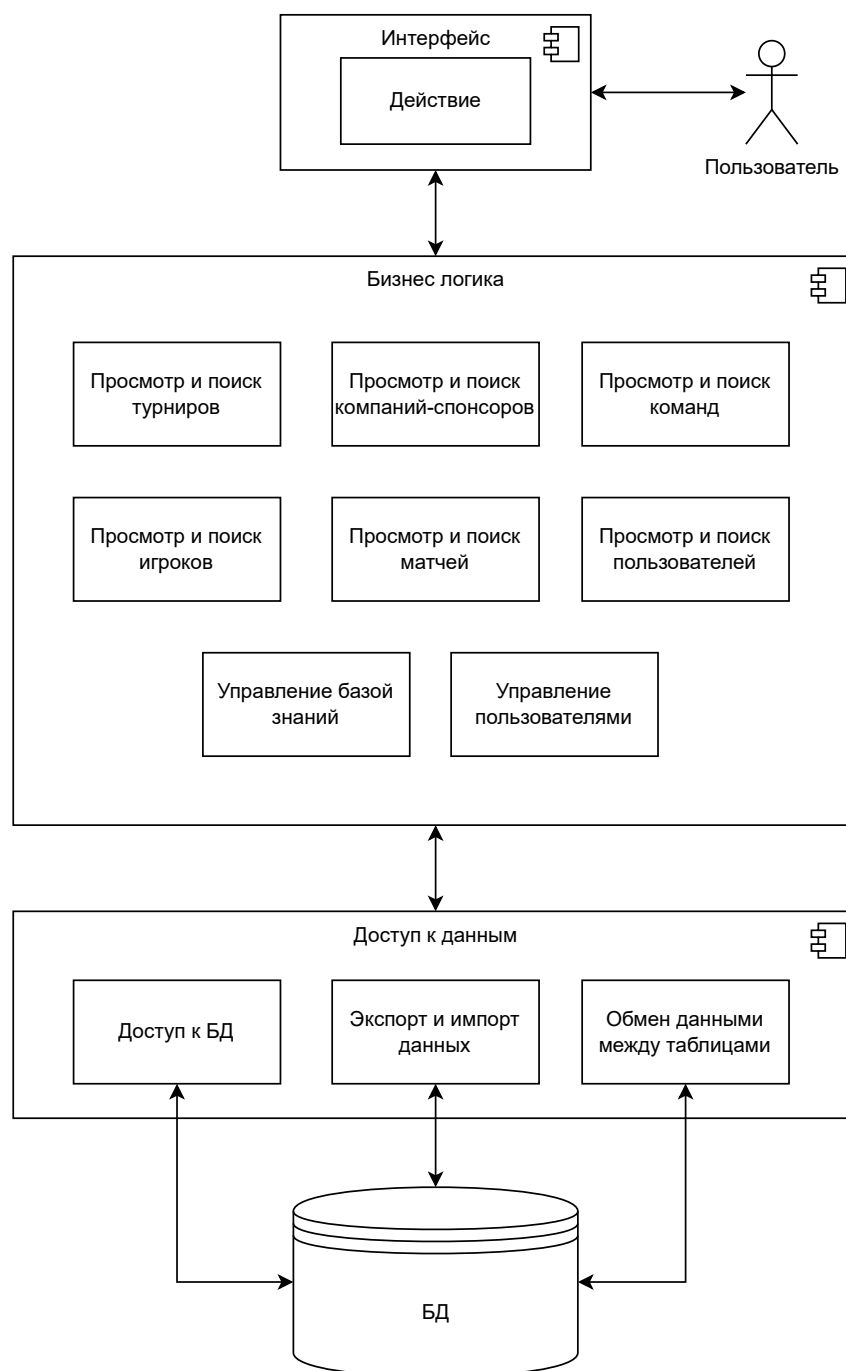


Рисунок 3.1 – Высокоуровневое разбиение на компоненты

3.3 Описание интерфейса программного продукта

При запуске приложения пользователю предоставляется возможность регистрации и создания собственного аккаунта в системе или выполнить вход под уже имеющимся (рисунок 3.2).

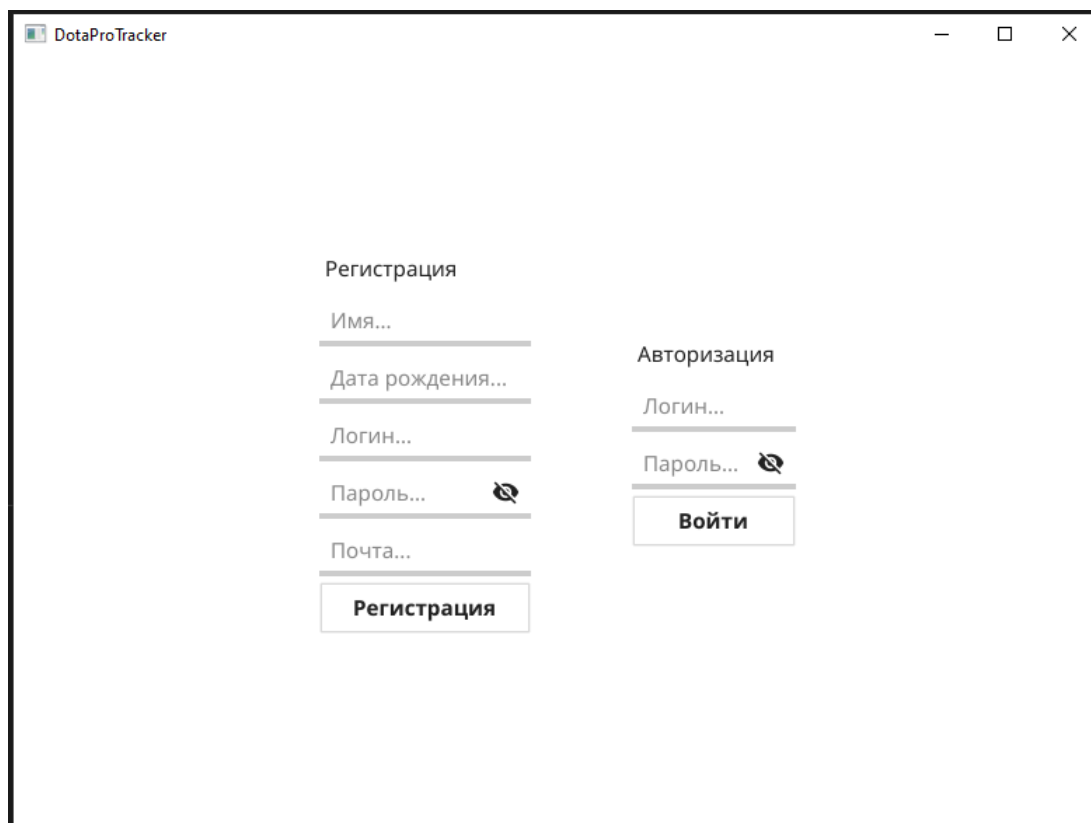


Рисунок 3.2 – Стартовое окно приложения

После выполнения процедуры идентификации происходит процедура аутентификации пользователя (выполняется проверка соответствия пароля введенного пользователем, с соответствующим паролем в базе данных, следует отметить, что пароли хранятся в зашифрованном виде с целью обеспечения сохранности и безопасности данных пользователей). В случае успешной аутентификации, пользователь получает доступ к функциям системы на основе своего уровня привилегий. На рисунке 3.3 изображен интерфейс администратора.

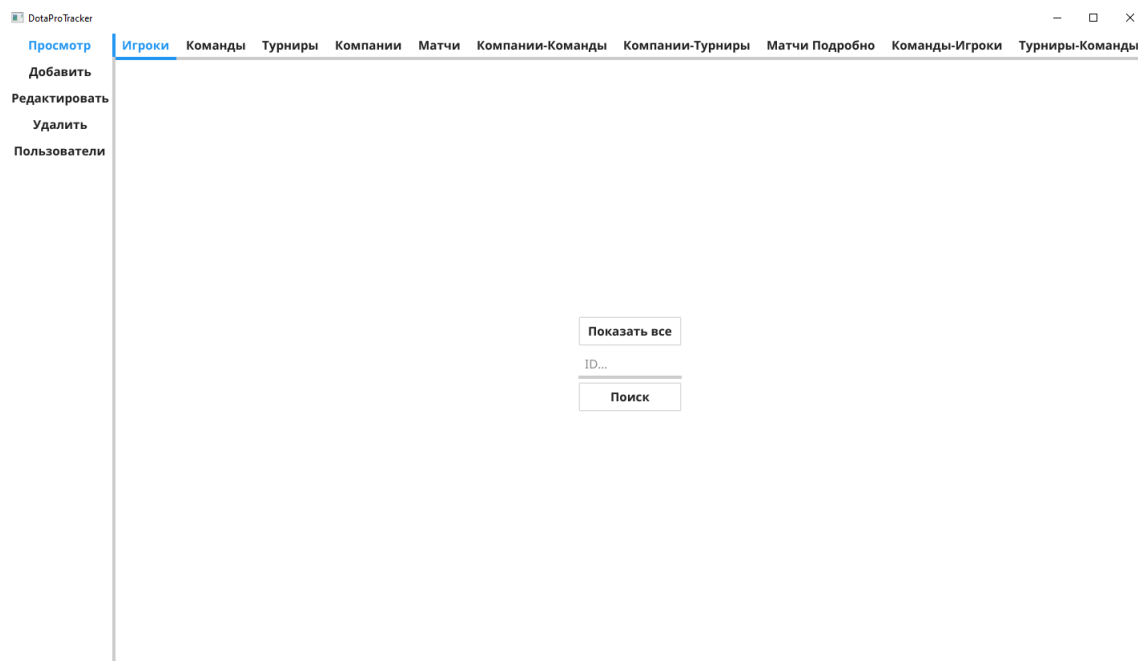


Рисунок 3.3 – Интерфейс администратора

Администратор системы имеет право добавлять новых пользователей в систему, а также смотреть весь список существующих аккаунтов (рисунки 3.4–3.5).

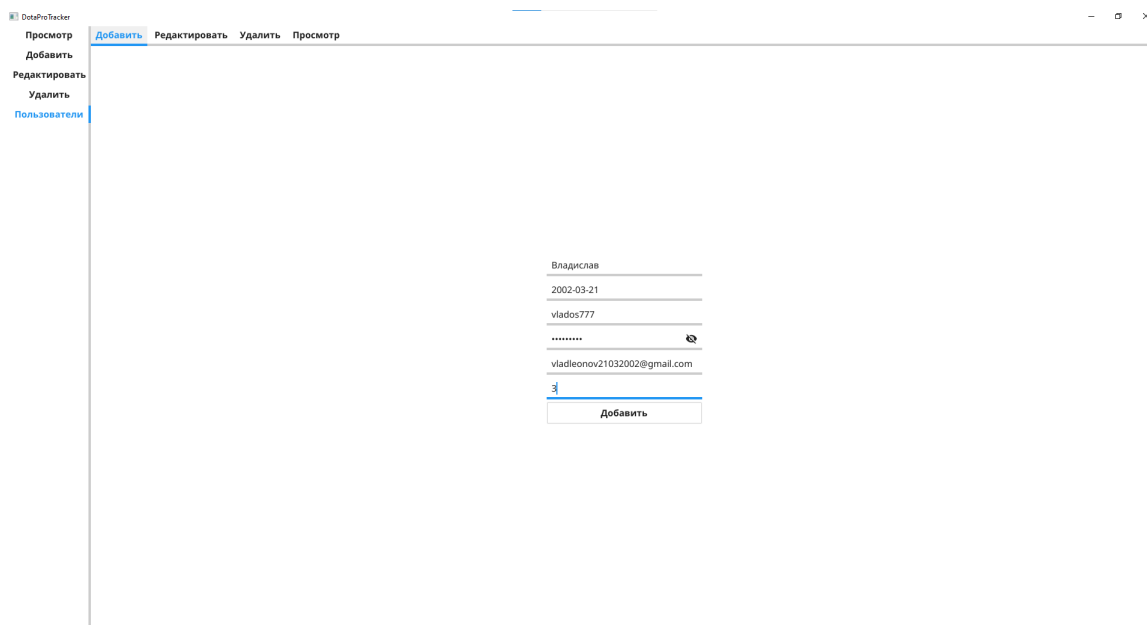


Рисунок 3.4 – Создание нового пользователя

Id	Name	Birthdate	Login	Email	PrivilegeLevel
1	admin	2001-01-02	admin	llooo	3
4	Владислав	2002-03-21	vlad05777	vladleonov21032002@gmail.com	3

Рисунок 3.5 – Список существующих пользователей

Роль модератора позволяет редактировать записи системы. Для навигации по записям реализована функция поиска (рисунки 3.6–3.7)

Игроки Команды **Турниры** Компании Матчи Компании-Команды Компании-Турниры Матчи Подробно Команды-Игроки Турниры-Команды

1
The International 10
1
20934393
2021-10-10
20
5000
Бухарест
Изменить

Рисунок 3.6 – Редактирование записи о турнире

Id	Name	Tier	PrizePool	DateStart	Duration	DPCPoints
1	The International 10	1	20934393	2021-10-10	20	5000

Рисунок 3.7 – Поиск записи о турнире

Обычный пользователь имеет возможность просмотра и поиска записей (рисунок 3.8).

Id	Name	Country	Website	Revenue	Industry
1	Womanly Macaw Inc.	Vatican City	womanlymacawinc.com	931200	Step-Aunt
2	Decorous Nickel Team	Kosovo	decorousnickelteam.com	351800	Act
3	Tense Pinafore Group	Trinidad & Tobago	tensepinaforegroup.com	162800	Solidity
4	Therapeutic Digit Company	China	therapeuticdigitcompany.com	380900	Harvester
5	Grubby Executor International	Cyprus	grubbyexecutorinternational.com	287500	Stump
6	Righteous Villa Team	Nicaragua	righteousvillateam.com	420400	Fraudster
7	Swanky Pate Team	Bangladesh	swankypateteam.com	344900	Quicksand
8	Swift Investigator Company	Bhutan	swiftinvestigatorcompany.com	273200	Approval
9	Disgusted Resist Group	Brunei	disgustedresistgroup.com	525200	Info
10	Amused Spelling Company	Palau	amusedspellingcompany.com	650800	Provider
11	Gigantic Transaction International	Morocco	gigantictransactioninternational.com	846000	Odyssey
12	Mute Carry Inc.	Zambia	mutecarryinc.com	438300	Parking
13	Nasty Redirect Group	China	nastyredirectgroup.com	908900	Overcharge
14	Cooling Filly Group	Belgium	coolingfillygroup.com	230900	Wraparound
15	Short Contributor Group	Canada	shortcontributorgroup.com	986300	Hold
16	Incandescent Poisoning Company	Qatar	incandescentpoisoningcompany.com	640000	Brochure
17	Broad Messy Team	Yemen	broadmessyteam.com	466200	Birch
18	Lean Rear Inc.	Taiwan	leanrearinc.com	973600	Proposition
19	Obsolete Crime Company	Japan	obsoletocrimecompany.com	35000	Marimba
20	Apathetic Sort Cooperation	Papua New Guinea	apatheticsortcooperation.com	513800	Beck
21	Temporary Bondsman Inc.	Norway	temporarybondsmaninc.com	634800	Gembok
22	Shaggy Noir Cooperation	Angola	shaggynoircooperation.com	861600	Basil

Рисунок 3.8 – Просмотр записей о компаниях-спонсорах

Вывод

В данном разделе были описаны выбранные технологии разработки, методология построения ПО, а также рассмотрено взаимодействие пользователя с приложением.”

4 Исследовательский раздел

Технические характеристики устройства, на котором выполнялось тестирование:

- Операционная система: Windows 10 64-bit [10].
- Память: 16 GB.
- Процессор: AMD Ryzen 5 4600H [11] @ 3.00 GHz.

Тестирование проводилось на ноутбуке при включённом режиме производительности. Во время тестирования ноутбук был нагружен только системными процессами.

Предметом исследования является скорость выполнения запросов к базе данных в зависимости от использования/игнорирования процесса индексации записей в таблице методом бинарного дерева. Следует отметить, что запросы выполнялись на стороне базы данных в многократном количестве, после чего выполнялось усреднение полученных значений.

Индекс был создан по строковому полю `signature_hero` таблицы `players`. В качестве тестового запроса был выбран следующий:

```
select * from players where signature_hero = 'Riki';
```

В результате эксперимента были полученные значения, представленные в таблице 4.1.

Таблица 4.1 – Сравнительный анализ времени выполнения запросов

Количество записей	С индексированием	Без индексирования
100	0.025 ms	0.030 ms
500	0.065 ms	0.145 ms
2000	0.080 ms	0.475 ms
5000	0.108 ms	1.040 ms
10000	0.176 ms	1.976 ms

Вывод

На основе полученных экспериментальным путем данных можно сделать вывод о том, что использование индексации позволяет существенным образом ускорить выполнение запроса (в 11,22 раза для таблицы из 10000 записей).

Однако следует учитывать тот факт, что на сравнительно небольших выборках данных индексация не дает столько значимого прироста быстродействия системы, а операция индексации требует дополнительной подготовки данных, а также увеличивает объем необходимой памяти для хранения. Так для исследуемой таблицы при 10000 записях потребовалось 88 Кбайт памяти для хранения дополнительных индексов (порядка 10% от общей памяти, занимаемой таблицей).

ЗАКЛЮЧЕНИЕ

В рамках курсовой работы была изучена предметная область для разрабатываемого программного продукта, рассмотрены существующие сервисы, выделены их основные характеристики и проведено сравнение, были описаны типы пользователей и возможные варианты взаимодействия с системой, проведена формализация данных и сущностей.

Цель курсовой работы была достигнута в полном объеме: была изучена предметная область, были выполнены проектирование и реализация базы данных, содержащей информацию о киберспортивных матчах и командах в дисциплине Dota 2, были созданы соответствующие триггеры и проведено исследование влияния индексации записей на скорость выполнения запросов в базе данных.

Следует отметить, что разработанное ПО в дальнейшем будет доработано посредством добавления системы анализа новых матчей на основе предыдущих, а также возможности просмотра прямых трансляций различных онлайн событий, проходящих в рамках конкретных турниров.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Most Watched Games on Twitch [Электронный ресурс]. — Режим доступа: <https://twitchtracker.com/games> (дата обращения: 08.09.2022).
2. Tracing Team Spirit's road from Russia to the Aegis of Champions [Электронный ресурс]. — Режим доступа: <https://www.redbull.com/int-en/team-spirit-road-to-world-champions-ti10> (дата обращения: 01.09.2022).
3. What is Dota 2? [Электронный ресурс]. — Режим доступа: <https://www.hotspawn.com/dota2/guides/what-is-dota-2> (дата обращения: 14.09.2022).
4. Dota Pro Circuit [Электронный ресурс]. — Режим доступа: https://liquipedia.net/dota2/Dota_Pro_Circuit (дата обращения: 11.09.2022).
5. OpenDota [Электронный ресурс]. — Режим доступа: <https://blog.opendota.com/2014/08/01/faq/> (дата обращения: 11.09.2022).
6. PostgreSQL: Documentation [Электронный ресурс]. — Режим доступа: <https://www.postgresql.org/docs/current/index.html> (дата обращения: 21.09.2022).
7. The Go Programming Language: Documentation [Электронный ресурс]. — Режим доступа: <https://go.dev/doc/> (дата обращения: 21.09.2022).
8. Fyne [Электронный ресурс]. — Режим доступа: <https://fyne.io/> (дата обращения: 21.09.2022).
9. The Clean Architecture [Электронный ресурс]. — Режим доступа: <https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html> (дата обращения: 21.09.2022).
10. Explore Windows 10. [Электронный ресурс]. — Режим доступа: <https://www.microsoft.com/en-us/windows/> (дата обращения: 02.10.2022).
11. AMD Processors [Электронный ресурс]. — Режим доступа: <https://www.amd.com/en/products/apu/amd-ryzen-5-4600h> (дата обращения: 02.10.2022).

ПРИЛОЖЕНИЕ А

Создание базы данных

Далее в листингах А.1–А.3 приведен сценарий создания базы данных.

Листинг А.1 – Сценарий создания базы данных (часть 1)

```
1      drop table if exists players cascade;
2      drop table if exists teams cascade;
3      drop table if exists tournaments cascade;
4      drop table if exists companies cascade;
5      drop table if exists teams_players cascade;
6      drop table if exists companies_teams cascade;
7      drop table if exists companies_tournaments cascade;
8      drop table if exists tournaments_teams cascade;
9      drop table if exists matches cascade;
10     drop table if exists match_performances cascade;
11     drop table if exists users cascade;
12
13     create table "players" (
14         "id" serial ,
15         "nickname" varchar ,
16         "realname" varchar ,
17         "birthdate" date ,
18         "country" varchar ,
19         "mmr" int ,
20         "role" varchar ,
21         "signature_hero" varchar
22     );
23
24     create table "teams" (
25         "id" serial ,
26         "name" varchar ,
27         "created_at" date ,
28         "email" varchar ,
29         "total_earnings" int ,
30         "region" varchar ,
31         "tier" int
32     );
```

Листинг А.2 – Сценарий создания базы данных (часть 2)

```
1      create table "tournaments" (  
2      "id" serial ,  
3      "name" varchar ,  
4      "tier" int ,  
5      "prize_pool" int ,  
6      "date_start" date ,  
7      "duration" int ,  
8      "dpc_points" int ,  
9      "location" varchar  
10     );  
11     create table "companies" (  
12     "id" serial ,  
13     "name" varchar ,  
14     "country" varchar ,  
15     "website" varchar ,  
16     "revenue" int ,  
17     "industry" varchar  
18     );  
19     create table "teams_players" (  
20     "id" serial ,  
21     "team_id" int ,  
22     "player_id" int ,  
23     "contract_date" date ,  
24     "contract_time" int  
25     );  
26     create table "companies_tournaments" (  
27     "id" serial ,  
28     "company_id" int ,  
29     "tournament_id" int ,  
30     "deposit" int  
31     );  
32     create table "companies_teams" (  
33     "id" serial ,  
34     "company_id" int ,  
35     "team_id" int ,  
36     "contract_date" date ,  
37     "contract_time" int  
38     );
```

Листинг А.3 – Сценарий создания базы данных (часть 3)

```
1      create table "tournaments_teams" (  
2      "id" serial ,  
3      "tournament_id" int ,  
4      "team_id" int ,  
5      "participation_type" varchar ,  
6      "is_winner" boolean  
7      );  
8      create table "matches" (  
9      "id" serial ,  
10     "tournament_id" int ,  
11     "r_team_id" int ,  
12     "d_team_id" int ,  
13     "duration" int ,  
14     "winner" boolean  
15     );  
16     create table "users" (  
17     "id" serial ,  
18     "name" varchar ,  
19     "birthdate" date ,  
20     "login" varchar ,  
21     "password" varchar ,  
22     "email" varchar ,  
23     "privilege_level" int  
24     );  
25     create table "match_performances" (  
26     "id" serial ,  
27     "match_id" int ,  
28     "player_id" int ,  
29     "team" boolean ,  
30     "hero" varchar ,  
31     "kills" int ,  
32     "deaths" int ,  
33     "assists" int ,  
34     "networth" int ,  
35     "gpm" int ,  
36     "xpm" int ,  
37     "dmg" int ,  
38     "heal" int ,  
39     "bld" int  
40     );
```

ПРИЛОЖЕНИЕ Б

Ограничения в базе данных

Далее в листингах Б.1–Б.4 приведен сценарий создания ограничений в базе данных.

Листинг Б.1 – Сценарий создания ограничений в базе данных (часть 1)

```
1      alter table "players" add constraint "players_id" primary key ("id");
2      alter table "teams" add constraint "teams_id" primary key ("id");
3      alter table "tournaments" add constraint "tournaments_id" primary key
      ("id");
4      alter table "companies" add constraint "companies_id" primary key
      ("id");
5      alter table "teams_players" add constraint "teams_players_id" primary
      key ("id");
6      alter table "companies_tournaments" add constraint
      "companies_tournaments_id" primary key ("id");
7      alter table "companies_teams" add constraint "companies_teams_id"
      primary key ("id");
8      alter table "tournaments_teams" add constraint "tournaments_teams_id"
      primary key ("id");
9      alter table "matches" add constraint "matches_id" primary key ("id");
10     alter table "users" add constraint "users_id" primary key ("id");
11     alter table "match_performances" add constraint "matches_performances_id"
      primary key ("id");
12     alter table "teams_players" add foreign key ("team_id") references
      "teams" ("id") on delete cascade;
13     alter table "teams_players" add foreign key ("player_id") references
      "players" ("id") on delete cascade;
14     alter table "companies_tournaments" add foreign key ("company_id")
      references "companies" ("id") on delete cascade;
15     alter table "companies_tournaments" add foreign key ("tournament_id")
      references "tournaments" ("id") on delete cascade;
16     alter table "companies_teams" add foreign key ("company_id") references
      "companies" ("id") on delete cascade;
17     alter table "companies_teams" add foreign key ("team_id") references
      "teams" ("id") on delete cascade;
18     alter table "tournaments_teams" add foreign key ("tournament_id")
      references "tournaments" ("id") on delete cascade;
19     alter table "tournaments_teams" add foreign key ("team_id") references
      "teams" ("id") on delete cascade;
20     alter table "matches" add foreign key ("tournament_id") references
      "tournaments" ("id") on delete cascade;
21     alter table "matches" add foreign key ("r_team_id") references "teams"
      ("id") on delete cascade;
22     alter table "matches" add foreign key ("d_team_id") references "teams"
      ("id") on delete cascade;
```

Листинг Б.2 – Сценарий создания ограничений в базе данных (часть 2)

```
1      alter table "match_performances" add foreign key ("match_id") references
      "matches" ("id") on delete cascade;
2      alter table "match_performances" add foreign key ("player_id")
      references "players" ("id") on delete cascade;
3
4      alter table "players" alter column "nickname" set not null;
5      alter table "players" alter column "realname" set not null;
6      alter table "players" alter column "birthdate" set not null;
7      alter table "players" alter column "country" set not null;
8      alter table "players" alter column "mmr" set not null;
9      alter table "players" alter column "role" set not null;
10     alter table "players" alter column "signature_hero" set not null;
11     alter table "players" add constraint "players_mmr_check" check (mmr >
      0);
12
13     alter table "teams" alter column "name" set not null;
14     alter table "teams" alter column "created_at" set not null;
15     alter table "teams" alter column "email" set not null;
16     alter table "teams" alter column "total_earnings" set not null;
17     alter table "teams" alter column "region" set not null;
18     alter table "teams" alter column "tier" set not null;
19     alter table "teams" add constraint "teams_total_earnings_check" check
      (total_earnings > 0);
20     alter table "teams" add constraint "teams_tier_check" check (tier > 0
      and tier < 5);
21
22     alter table "tournaments" alter column "name" set not null;
23     alter table "tournaments" alter column "tier" set not null;
24     alter table "tournaments" alter column "prize_pool" set not null;
25     alter table "tournaments" alter column "date_start" set not null;
26     alter table "tournaments" alter column "duration" set not null;
27     alter table "tournaments" alter column "dpc_points" set not null;
28     alter table "tournaments" alter column "location" set not null;
29     alter table "tournaments" add constraint "tournaments_tier_check" check
      (tier > 0 and tier < 5);
30     alter table "tournaments" add constraint
      "tournaments_prize_pool_check" check (prize_pool > 0);
31     alter table "tournaments" add constraint "tournaments_date_check" check
      (duration > 0);
32     alter table "tournaments" add constraint
      "tournaments_dpc_points_check" check (dpc_points >= 0);
```

Листинг Б.3 – Сценарий создания ограничений в базе данных (часть 3)

```
1      alter table "companies" alter column "name" set not null;
2      alter table "companies" alter column "country" set not null;
3      alter table "companies" alter column "website" set not null;
4      alter table "companies" alter column "revenue" set not null;
5      alter table "companies" alter column "industry" set not null;
6      alter table "companies" add constraint "companies_revenue" check
      (revenue > 0);
7
8      alter table "teams_players" alter column "team_id" set not null;
9      alter table "teams_players" alter column "player_id" set not null;
10     alter table "teams_players" alter column "contract_date" set not null;
11     alter table "teams_players" alter column "contract_time" set not null;
12     alter table "teams_players" add constraint
      "teams_players_contract_time_check" check (contract_time > 0 and
      contract_time <= 36);
13
14     alter table "companies_tournaments" alter column "company_id" set not
      null;
15     alter table "companies_tournaments" alter column "tournament_id" set
      not null;
16     alter table "companies_tournaments" alter column "deposit" set not null;
17     alter table "companies_tournaments" add constraint
      "companies_tournaments_deposit_check" check (deposit > 0);
18
19     alter table "companies_teams" alter column "company_id" set not null;
20     alter table "companies_teams" alter column "team_id" set not null;
21     alter table "companies_teams" alter column "contract_date" set not null;
22     alter table "companies_teams" alter column "contract_time" set not null;
23     alter table "companies_teams" add constraint
      "companies_teams_contract_time_check" check (contract_time > 0 and
      contract_time <= 36);
24
25     alter table "tournaments_teams" alter column "tournament_id" set not
      null;
26     alter table "tournaments_teams" alter column "team_id" set not null;
27     alter table "tournaments_teams" alter column "participation_type" set
      not null;
28     alter table "tournaments_teams" alter column "is_winner" set not null;
29     alter table "tournaments_teams" add constraint
      "tournaments_teams_participation_type_check" check
      (participation_type in ('invite', 'qualification'));
```

Листинг Б.4 – Сценарий создания ограничений в базе данных (часть 4)

```

1      alter table "matches" alter column "tournament_id" set not null;
2      alter table "matches" alter column "r_team_id" set not null;
3      alter table "matches" alter column "d_team_id" set not null;
4      alter table "matches" alter column "duration" set not null;
5      alter table "matches" alter column "winner" set not null;
6      alter table "matches" alter column "matches_duration_check" check
      (duration > 0);
7      alter table "users" alter column "name" set not null;
8      alter table "users" alter column "birthdate" set not null;
9      alter table "users" alter column "login" set not null;
10     alter table "users" alter column "password" set not null;
11     alter table "users" alter column "email" set not null;
12     alter table "users" alter column "privilege_level" set not null;
13     alter table "users" add constraint "users_login" unique (login);
14     alter table "users" add constraint "users_lvl_check" check
      (privilege_level > 0 and privilege_level < 4);
15     alter table "match_performances" alter column "match_id" set not null;
16     alter table "match_performances" alter column "player_id" set not null;
17     alter table "match_performances" alter column "team" set not null;
18     alter table "match_performances" alter column "hero" set not null;
19     alter table "match_performances" alter column "kills" set not null;
20     alter table "match_performances" alter column "deaths" set not null;
21     alter table "match_performances" alter column "assists" set not null;
22     alter table "match_performances" alter column "networth" set not null;
23     alter table "match_performances" alter column "gpm" set not null;
24     alter table "match_performances" alter column "xpm" set not null;
25     alter table "match_performances" alter column "dmg" set not null;
26     alter table "match_performances" alter column "heal" set not null;
27     alter table "match_performances" alter column "bld" set not null;
28     alter table "match_performances" add constraint
      "match_performances_kills_check" check (kills >= 0);
29     alter table "match_performances" add constraint
      "match_performances_deaths_check" check (deaths >= 0);
30     alter table "match_performances" add constraint
      "match_performances_assists_check" check (assists >= 0);
31     alter table "match_performances" add constraint
      "match_performances_networth_check" check (networth > 0);
32     alter table "match_performances" add constraint
      "match_performances_gpm_check" check (gpm > 0);
33     alter table "match_performances" add constraint
      "match_performances_xpm_check" check (xpm > 0);
34     alter table "match_performances" add constraint
      "match_performances_dmg_check" check (dmg >= 0);
35     alter table "match_performances" add constraint
      "match_performances_heal_check" check (heal >= 0);
36     alter table "match_performances" add constraint
      "match_performances_bld_check" check (bld >= 0);

```

ПРИЛОЖЕНИЕ В

Ролевая модель

Далее в листингах В.1–В.3 приведен сценарий создания ролевой модели в базе данных.

Листинг В.1 – Сценарий создания ролевой модели в базе данных (часть 1)

```
1      do
2      $do$
3      begin
4      if exists (
5      select from pg_catalog.pg_roles
6      where rolname = 'initial_login') then
7      raise notice 'role "initial_login" already exists. skipping.';
8      else
9      create role initial_login with nosuperuser nocreatedb login password
10         'initial_login';
11      end if;
12      end
13      $do$;
14      do
15      $do$
16      begin
17      if exists (
18      select from pg_catalog.pg_roles
19      where rolname = 'default_user') then
20      raise notice 'role "default_user" already exists. skipping.';
21      else
22      create role default_user with nosuperuser nocreatedb login password
23         'default_user';
24      end if;
25      end
26      $do$;
27      do
28      $do$
29      begin
30      if exists (
31      select from pg_catalog.pg_roles
32      where rolname = 'moderator') then
33      raise notice 'role "moderator" already exists. skipping.';
34      else
35      create role moderator with nosuperuser nocreatedb login password
36         'moderator';
37      end if;
38      end
39      $do$;
```


Листинг В.2 – Сценарий создания ролевой модели в базе данных (часть 2)

```
1      do
2      $do$
3      begin
4      if exists (
5      select from pg_catalog.pg_roles
6      where rolname = 'administrator') then
7      raise notice 'role "administrator" already exists. skipping.';
8      else
9      create role administrator with nosuperuser nocreatedb login password
        'administrator';
10     end if;
11     end
12     $do$;
13
14     grant select , insert on table users to initial_login;
15     grant usage, select on sequence users_id_seq to initial_login;
16
17     grant select on table players to default_user;
18     grant select on table teams to default_user;
19     grant select on table tournaments to default_user;
20     grant select on table companies to default_user;
21     grant select on table teams_players to default_user;
22     grant select on table companies_teams to default_user;
23     grant select on table companies_tournaments to default_user;
24     grant select on table tournaments_teams to default_user;
25     grant select on table matches to default_user;
26     grant select on table match_performances to default_user;
27
28     grant select , insert , delete , update on table players to moderator;
29     grant select , insert , delete , update on table teams to moderator;
30     grant select , insert , delete , update on table tournaments to moderator;
31     grant select , insert , delete , update on table companies to moderator;
32     grant select , insert , delete , update on table teams_players to
        moderator;
33     grant select , insert , delete , update on table companies_teams to
        moderator;
34     grant select , insert , delete , update on table companies_tournaments to
        moderator;
35     grant select , insert , delete , update on table tournaments_teams to
        moderator;
36     grant select , insert , delete , update on table matches to moderator;
37     grant select , insert , delete , update on table match_performances to
        moderator;
```

Листинг В.3 – Сценарий создания ролевой модели в базе данных (часть 3)

```
1      grant usage, select on sequence players_id_seq to moderator;
2      grant usage, select on sequence teams_id_seq to moderator;
3      grant usage, select on sequence tournaments_id_seq to moderator;
4      grant usage, select on sequence companies_id_seq to moderator;
5      grant usage, select on sequence teams_players_id_seq to moderator;
6      grant usage, select on sequence companies_teams_id_seq to moderator;
7      grant usage, select on sequence companies_tournaments_id_seq to
      moderator;
8      grant usage, select on sequence tournaments_teams_id_seq to moderator;
9      grant usage, select on sequence matches_id_seq to moderator;
10     grant usage, select on sequence match_perfomances_id_seq to moderator;
11
12     grant select, insert, delete, update on table players to administrator;
13     grant select, insert, delete, update on table teams to administrator;
14     grant select, insert, delete, update on table tournaments to
      administrator;
15     grant select, insert, delete, update on table companies to
      administrator;
16     grant select, insert, delete, update on table teams_players to
      administrator;
17     grant select, insert, delete, update on table companies_teams to
      administrator;
18     grant select, insert, delete, update on table companies_tournaments to
      administrator;
19     grant select, insert, delete, update on table tournaments_teams to
      administrator;
20     grant select, insert, delete, update on table matches to administrator;
21     grant select, insert, delete, update on table match_perfomances to
      administrator;
22     grant select, insert, delete, update on table users to administrator;
23
24     grant usage, select on sequence players_id_seq to administrator;
25     grant usage, select on sequence teams_id_seq to administrator;
26     grant usage, select on sequence tournaments_id_seq to administrator;
27     grant usage, select on sequence companies_id_seq to administrator;
28     grant usage, select on sequence teams_players_id_seq to administrator;
29     grant usage, select on sequence companies_teams_id_seq to administrator;
30     grant usage, select on sequence companies_tournaments_id_seq to
      administrator;
31     grant usage, select on sequence tournaments_teams_id_seq to
      administrator;
32     grant usage, select on sequence matches_id_seq to administrator;
33     grant usage, select on sequence match_perfomances_id_seq to
      administrator;
34     grant usage, select on sequence users_id_seq to administrator;
```

ПРИЛОЖЕНИЕ Г

Триггеры

Далее в листингах Г.1–Г.2 приведен сценарий создания триггеров в базе данных.

Листинг Г.1 – Сценарий создания триггеров в базе данных (часть 1)

```
1      create or replace function make_deposit()
2      returns trigger
3      as
4      $$
5      begin
6      update tournaments
7      set prize_pool = prize_pool + new.deposit
8      where id = new.tournament_id;
9      return new;
10     end;
11     $$
12     language plpgsql;
13
14     drop trigger if exists make_deposit_trigger on companies_tournaments;
15     create trigger make_deposit_trigger after insert on
16         companies_tournaments
17         for row execute procedure make_deposit();
```

Листинг Г.2 – Сценарий создания триггеров в базе данных (часть 2)

```
1      create or replace function make_win()
2      returns trigger
3      as
4      $$
5      declare
6      tmp int = 0;
7      begin
8      if (new.is_winner = true) then
9      select into tmp prize_pool from tournaments_teams join tournaments
10     on tournaments.id = tournaments_teams.id
11     where tournaments.id = new.tournament_id;
12
13     update teams
14     set total_earnings = total_earnings + tmp
15     where teams.id = new.team_id;
16     end if;
17     return new;
18     end;
19     $$
20     language plpgsql;
21
22     drop trigger if exists make_win on tournaments_teams;
23     create trigger make_win after insert on tournaments_teams
24     for row execute procedure make_win();
```